UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/723,329	11/26/2003	Mikhail Arkhipov	13768.466	9473

47973   7590   05/16/2007

WORKMAN NYDEGGER/MICROSOFT
1000 EAGLE GATE TOWER
60 EAST SOUTH TEMPLE
SALT LAKE CITY, UT 84111

EXAMINER
RUTLEDGE, AMELIA L

ART UNIT	PAPER NUMBER
2176	

MAIL DATE	DELIVERY MODE
05/16/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

-- *The MAILING DATE of this communication appears on the cover sheet with the correspondence address* --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE <u>3</u> MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1) ☒ Responsive to communication(s) filed on <u>14 March 2007</u>.

2a) ☒ This action is **FINAL**.   2b) ☐ This action is non-final.

3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4) ☒ Claim(s) <u>1,3-15,25 and 27-37</u> is/are pending in the application.

    4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5) ☐ Claim(s) _____ is/are allowed.

6) ☒ Claim(s) <u>1,3-15,25 and 27-37</u> is/are rejected.

7) ☐ Claim(s) _____ is/are objected to.

8) ☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9) ☐ The specification is objected to by the Examiner.

10) ☐ The drawing(s) filed on _____ is/are: a) ☐ accepted or b) ☐ objected to by the Examiner.

    Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

    Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

    a) ☐ All   b) ☐ Some * c) ☐ None of:

      1. ☐ Certified copies of the priority documents have been received.

      2. ☐ Certified copies of the priority documents have been received in Application No. _____.

      3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

    * See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)

2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☐ Information Disclosure Statement(s) (PTO/SB/08)
    Paper No(s)/Mail Date _____.

4) ☐ Interview Summary (PTO-413)
    Paper No(s)/Mail Date. _____.

5) ☐ Notice of Informal Patent Application

6) ☐ Other: _____.

DETAILED ACTION

1.   This action is responsive to communications: Amendment, filed 03/14/2007.

2.   Claims 1, 3-15, 25, and 27-37 are pending in the case. Claims 1, 9, 25, and 32

are independent claims.

3.   The Amendments to the Specification filed 03/14/2007 will be entered.

Claim Rejections - 35 USC § 112

1.   The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly
> claiming the subject matter which the applicant regards as his invention.

2.   Claims 3-6 and 27-29 are rejected under 35 U.S.C. 112, second paragraph, as

being indefinite for failing to particularly point out and distinctly claim the subject matter

which applicant regards as the invention.

3.   Claims 3-6 depend from dependent claim 2, which has been canceled, therefore

there is insufficient antecedent basis for the limitations in the claim. For purposes of

examination, claims 3-6 are interpreted as depending from independent claim 1.

Appropriate correction is required.

4.   Claims 27-29 depend from dependent claim 26, which has been canceled,

therefore there is insufficient antecedent basis for the limitations in the claim. For

purposes of examination, claims 27-29 are interpreted as depending from independent

claim 25. Appropriate correction is required.

Claim Rejections - 35 USC § 103

4.   The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all

obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

5.   **Claims 1, 3-15, 25, and 27-37 are rejected under 35 U.S.C. 103(a) as being**

unpatentable over Chan et al. (hereinafter "Chan"), U.S. Patent No. 6,799,718

issued October 2004, in view of NetBeans: The Definitive Guide (hereinafter

"NetBeans"), Bodreau, et. al., published by O'Reilly, October 2002, p. 1-12 printed

from http://proquest.safaribooksonline.com/0596002807.

        Regarding independent claim 1, Chan teaches an integrated development

environment (IDE) for editing multilanguage documents (Abstract). Chan teaches

accessing and editing a multilanguage file with a plurality of segments written in

different programming languages (Col. 2, l. 52-Col. 3, l. 63), using secondary scanner

and parser programs (Col. 7, l. 31-50 and Col. 8, l. 1-25). Chan teaches presenting the

multilanguage file in a primary application view that includes the plurality of code

segments (Fig. 5; Col. 8, l. 60-Col. 9, l. 56). Chan teaches enabling a user to edit the

different segments from within the primary view, and teaches a text editor and user

interface integrated with the primary scanner and at least one supplemental scanner

(claims 31 and 32). Chan teaches a system for performing and tracking edits, and

providing advanced editing functions, such as error detection, for sections of code for

each of the various languages using secondary editors including parsers, scanners,

converters, buffers, and engines (claims 10, 19, 20, 28, 34) to perform the editing

functions for each language, mapped back to the main file M of the primary view (col. 8,

l. 60-col. 9, l. 56; Fig. 5); compare to *without requiring the programmer to leave the*

primary application view to open or interface with the secondary editors, enabling the

programmer to edit the different code segments of the multilanguage document from

within the primary application view by editing code segments written in the primary

language with the primary editor, and

by sending the at least one other code segment written in secondary

programming language to one of the secondary editors so that thereafter, edits made in

the primary application view will be performed by the secondary editor even though the

programmer is working on the multilanguage document only in the primary application

view.

Chan discloses creating corresponding secondary files to be used to edit the at

least one other code segment, the code segments in other languages, (Fig. 5, col. 9, l.

15-64). Chan teaches a buffer coordinator recognizing edits made in the primary

application view to the at least one other code segment and transmitting the edits to the

secondary editor, since Chan discloses that each section of code for each language is

provided to the appropriate accumulator to be placed in a buffer for code of the same

language, and the main file M is marked to indicate the section of code converted or

accumulated (col. 7, l. 51-col. 8, l. 30; col. 9, l. 5-16). Chan teaches that the buffer

coordinator replicates changes from the secondary file to the primary document (col. 7,

l. 51-col. 8, l. 30; col. 8, l. 31-59), since all errors and changes are mapped back to the

mail file M.

While Chan teaches secondary scanner and parser programs, integrated with the

text editor and user interface, Chan does not explicitly teach editing the other code

segment written in the secondary programming language through interaction with the

corresponding secondary editor, and displaying the other code segment written in the

secondary programming language and the edits made within the primary application

view and within the multilanguage document in which the other code segment is

contained. However, NetBeans teaches editing code segments in secondary

programming languages through interaction with secondary editors, and displaying the

edits in the primary application view (Sect 3.11 "Other Editor Functionality"; Appendix A:

A.1.1. "The XML Modules" and Appendix A: A.1.2. "The editor"; A.1.13. "Socket-Based

Editor Support-the External Editor Module").

NetBeans was an Open Source IDE which contained editing modules and

allowed programmers to extend and add on editing modules with advanced editing

functionality for different languages, as discussed in Appendix A (A.1.1. "The XML

Modules"; A: A.1.2. "The editor"; A.1.13. "Socket-Based Editor Support-the External

Editor Module"). NetBeans taught two ways to open a file in the NetBeans Source

Editor, either by the Open action, which would bring up a special editor, or the Edit

action which would open the NetBeans Source Editor (Sect. 3.2 "Opening the Source

Editor"). NetBeans discloses that the Source Editor edits HTML, XML, Java Properties

files, and other programming languages as NetBeans core developers and external

open source contributors add modules to handle those languages (Sect. 3.11. "Other

Editor Functionality"). Therefore NetBeans teaches editing the other code segment

written in the secondary programming language through interaction with the

corresponding secondary editor, and displaying the other code segment written in the

secondary programming language and the edits made within the primary application

view and within the multilanguage document in which the other code segment is

contained. Appendix A discloses how the editing functionality was added to the

secondary editing modules. NetBeans also discloses a secondary editor creating a

corresponding secondary file to be used to edit the at least one other code segment,

since NetBeans teaches creating new JSP and HTML files (p. 6).

Both Chan and NetBeans are directed toward multilanguage IDEs having

advanced editing functionality. Both inventions teach editing different programming

languages through a primary application view of a primary editor. It would have been

obvious to one of ordinary skill in the art at the time of the invention to apply the

modular, open source and extensible IDE of NetBeans having editing modules for

different programming languages with advanced editing functionality to Chan, since

Chan teaches a modular approach to providing completion assistance for different

languages (Chan, col. 11, l. 45-col. 12, l. 48), including delegating sections of code to

different engines, a modular programming approach which would have made the

combination with the NetBeans editor modules both obvious and desirable.

Regarding dependent claim 3, Chan teaches that code for each language is
parsed and stored separately (Col. 9, l. 16-56; Col. 7, l. 31-50 and Col. 8, l. 1-25), i.e.,
replicating secondary code languages to a secondary document, and editing code
written in a language of the received code segment. Chan teaches a mapping between
the working files and the primary file (Col. 7, l. 31-50 and Col. 8, l. 1-25). Chan teaches
a system for performing and tracking edits, and providing advanced editing functions,
such as error detection, for sections of code for each of the various languages using
secondary editors including parsers, scanners, converters, buffers, and engines (claims
10, 19, 20, 28, 34) to perform the editing functions for each language, mapped back to
the main file M of the primary view (col. 8, l. 60-col. 9, l. 56; Fig. 5). Figure 5 shows how
the edits are replicated to the secondary documents.

Regarding dependent claim 4, Chan teaches that the Chan teaches that code
for each secondary language is parsed and stored separately (Col. 9, l. 16-56), in
documents that are transparent, i.e., not visible to the user (col. 8, l. 31-59).

Regarding dependent claim 5, Chan teaches that code for the secondary
documents may or may not require conversion from text format (Col. 7, l. 31-50 and Col.
8, l. 1-25) and that unconverted code is accumulated into a working file or buffer.

Regarding dependent claim 6, Chan teaches a mapping between working files
and the primary file (Col. 7, l. 31-50 and Col. 8, l. 1-25; Fig. 5), and such a mapping

would enable the primary editor to modify the multilanguage file in response to a change made to the secondary document, since the files were linked and updated.

Regarding dependent claim 7, Chan teaches that the primary file may be modified in response to advanced editing features of the secondary editor that are not inherently enabled by the primary editor, for example, using a Java Completion engine to modify code segments (Col. 12, l. 29-49).

Regarding dependent claim 8, Chan teaches syntax coloring (Col. 5, l. 64-65).

Regarding independent claim 9 Chan teaches an integrated development environment (IDE) for editing multilanguage documents (Abstract). Chan teaches accessing and editing a multilanguage file with a plurality of segments written in different programming languages (Col. 2, l. 52-Col. 3, l. 63), using secondary scanner and parser programs (Col. 7, l. 31-50 and Col. 8, l. 1-25). Chan teaches presenting the multilanguage file in a primary application view that includes the plurality of code segments (Fig. 5; Col. 8, l. 60-Col. 9, l. 56). Chan teaches enabling a user to edit the different segments from within the primary view, and teaches a text editor and user interface integrated with the primary scanner and at least one supplemental scanner (claims 31 and 32). Chan teaches a system for performing and tracking edits, and providing advanced editing functions, such as error detection, for sections of code for each of the various languages using secondary editors including parsers, scanners,

converters, buffers, and engines (claims 10, 19, 20, 28, 34) to perform the editing

functions for each language, mapped back to the main file M of the primary view (col. 8,

l. 60-col. 9, l. 56; Fig. 5).

Chan teaches a mapping between working files and the primary file (Col. 7, l. 31-

50 and Col. 8, l. 1-25; Fig. 5), and such a mapping would enable the primary editor to

modify the multilanguage file in response to a change made to the secondary

document, since the files were linked and updated. Chan teaches providing completion

assistance for both primary and secondary programming languages via the primary

view (col. 11, l. 46-col. 12, l. 29); compare to *identifying whether the at least one other*

code segment written in a secondary programming language is a complete code

segment, and if not, supplementing the at least one other code segments with additional

data necessary to create complete source code for the at least one other code segment,

so that it can be recognized and edited by the secondary editor for the language of the

at least one other code segment. Chan teaches that the primary editor modifies the

primary application view to accommodate for the editing functionality provided by the

secondary editors for the code segments in the secondary languages (col. 7, l. 17-51).

While Chan teaches secondary scanner and parser programs, integrated with the

text editor and user interface, Chan does not explicitly teach secondary editors having

advanced editing functionality not available within the primary editor. However,

NetBeans teaches secondary editors having advanced editing functionality not available

within the primary editor (Sect 3.11 "Other Editor Functionality"; Appendix A: A.1.1. "The

XML Modules" and Appendix A: A.1.2. "The editor"; A.1.13. "Socket-Based Editor

Support-the External Editor Module"), and which, without requiring the programmer to open or interface with the secondary editors, provide at least one other code segment to the secondary editor having advanced editing functionality for the language of that code segment. NetBeans further discloses a secondary editor creating a corresponding secondary file to be used to edit the at least one other code segment, since NetBeans teaches creating new JSP and HTML files (p. 6).

NetBeans was an Open Source IDE which contained editing modules and allowed programmers to extend and add on editing modules with advanced editing functionality for different languages, as discussed in Appendix A (A.1.1. "The XML Modules"; A: A.1.2. "The editor"; A.1.13. "Socket-Based Editor Support-the External Editor Module"). NetBeans taught two ways to open a file in the NetBeans Source Editor, either by the Open action, which would bring up a special editor, or the Edit action which would open the NetBeans Source Editor (Sect. 3.2 "Opening the Source Editor"). NetBeans discloses that the Source Editor edits HTML, XML, Java Properties files, and other programming languages as NetBeans core developers and external open source contributors add modules to handle those languages (Sect. 3.11. "Other Editor Functionality"). Therefore NetBeans teaches that the editing modules can be accessed through the Source Editor without requiring the programmer to open or interface with the secondary editors. Appendix A discloses how the advanced editing functionality was added to the editing modules.

Both Chan and NetBeans are directed toward multilanguage IDEs having advanced editing functionality. Both inventions teach editing different programming

languages through a primary application view of a primary editor. It would have been

obvious to one of ordinary skill in the art at the time of the invention to apply the

modular, open source and extensible IDE of NetBeans having editing modules for

different programming languages with advanced editing functionality to Chan, since

Chan teaches a modular approach to providing completion assistance for different

languages (Chan, col. 11, l. 45-col. 12, l. 48), including delegating sections of code to

different engines, a modular programming approach which would have made the

combination with the NetBeans modules both obvious and desirable.

Regarding dependent claim 10, Chan teaches a menu in the primary

application view created by the secondary completion engine, which is a completion

assistance menu (Fig. 1, Col. 5, l. 25-37; Col. 12, l. 1-16).

Regarding dependent claims 11 and 12, Chan teaches that code for each

language is parsed and stored separately (Col. 9, l. 16-56; Col. 7, l. 31-50 and Col. 8, l.

1-25), i.e., replicating secondary code languages to a secondary document. Chan

teaches a mapping between the working files and the primary file (Col. 7, l. 31-50 and

Col. 8, l. 1-25), and the mapping would enable the primary editor to replicate and modify

the multilanguage file in response to a change made to the secondary document, since

the files were linked. Chan teaches that the functionality provided by the secondary

editors are not available to the primary editor, since Chan teaches the use of registered

completion engines called from the IDE (col. 11, l. 46-col. 12, l. 67).

Regarding dependent claim 13, claim 13 is directed toward substantially similar subject matter as claimed in dependent claim 8, and is rejected along the same rationale.

Regarding dependent claims 14 and 15, Chan teaches a method of tracking the sections of code for the various languages by using mapping between buffers and the main file, i.e., primary document. The mapping function translates the structural and error information between the buffered code and the primary document (Col. 9, I. 16-63). It was inherent in the teaching of Chan that such a mapping would determine the validity edits and avoid replication, since the mapping enabled tracking of code position and changes in code between the primary file and the working files.

Regarding independent claim 25 and dependent claims 27-31, claims 25-31 reflect the computer program product used to implement the methods claimed in independent claim 1 and dependent claims 2-4 and 6-8, respectively, and are rejected along the same rationale.

Regarding independent claim 32 and dependent claims 33-36, claims 32-36 reflect the method and computer program product used to implement the methods claimed in independent claim 9 and dependent claims 10-12 and 13, respectively, and are rejected along the same rationale.

Regarding dependent claim 37, Chan teaches a method of tracking the

sections of code for the various languages by using mapping between buffers and the

main file, i.e., primary file, and the secondary files. The mapping function translates the

structural and error information between the buffered code and the primary document

(Col. 9, l. 16-63). It was inherent in the disclosure of Chan that such a mapping would

determine the validity edits and avoid an infinite loop of replication, since the mapping

enabled tracking of code position and changes in code between the primary document

and the working files.

Response to Arguments

5.   Applicant's arguments filed 03/14/2007 have been fully considered but they are

not persuasive.

Chan discloses creating corresponding secondary files to be used to edit the at

least one other code segment, the code segments in other languages, (Fig. 5, col. 9, l.

15-64). Chan teaches a buffer coordinator recognizing edits made in the primary

application view to the at least one other code segment and transmitting the edits to the

secondary editor, since Chan discloses that each section of code for each language is

provided to the appropriate accumulator to be placed in a buffer for code of the same

language, and the main file M is marked to indicate the section of code converted or

accumulated (col. 7, l. 51-col. 8, l. 30; col. 9, l. 5-16). Chan teaches that the buffer

coordinator replicates changes from the secondary file to the primary document (col. 7,

l. 51-col. 8, l. 30; col. 8, l. 31-59), since all errors and changes are mapped back to the mail file M.

Further, NetBeans also discloses a secondary editor creating a corresponding secondary file to be used to edit the at least one other code segment, since NetBeans teaches creating new JSP and HTML files (p. 6).

Therefore, the combination of Chan and NetBeans does teach each and every limitation of the claimed invention, and does disclose applicant's newly claimed limitations,

...*the secondary editor creating a corresponding secondary file to be used to edit the at least one other code segment,*

a buffer coordinator recognizing edits made in the primary application view to the at least one other code segment and transmitting the edits to the secondary editor....

...

the buffer coordinator replicating changes from the secondary file to the primary document... (Claim 1).

Conclusion

6.   **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not

mailed until after the end of the THREE-MONTH shortened statutory period, then the

shortened statutory period will expire on the date the advisory action is mailed, and any

extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of

the advisory action. In no event, however, will the statutory period for reply expire later

than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the

examiner should be directed to Amelia Rutledge whose telephone number is 571-272-

7508. The examiner can normally be reached on Monday - Friday 9:30 - 6:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's

supervisor, Heather Herndon can be reached on 571-272-4136. The fax phone number

for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the

Patent Application Information Retrieval (PAIR) system. Status information for

published applications may be obtained from either Private PAIR or Public PAIR.

Status information for unpublished applications is available through Private PAIR only.

For more information about the PAIR system, see http://pair-direct.uspto.gov. Should

you have questions on access to the Private PAIR system, contact the Electronic

Business Center (EBC) at 866-217-9197 (toll-free).

AR

Doug Hutton
Primary Examiner
Technology Center 2100